EMAIL CLASSIFICATION USING A SELF-LEARNING TECHNIQUE

BASED ON USER PREFERENCES

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Swapna Gautam Phadke

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

October 2015

Fargo, North Dakota

North Dakota State University
Graduate School

Title

Email Classification using a Self-Learning mechanism based on User Preferences

By

Swapna Gautam Phadke

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dr. Wei Jin
Chair

Dr. Saeed Salem

Dr. Na Gong

Approved:

11/05/2015
Date

Dr. Brian M Slator
Department Chair

ABSTRACT

In the current information overloaded atmosphere with an explosive growth of textual data, one can find it a challenging task to keep all the ducks in a row. This has resulted in an emergence of many Text Classification algorithms. Text classification is a process of categorizing data in pre-defined categories based on Topics or Genre. It is used - to classify named entities, Twitter and newspaper feeds, medical repository and Email.

In this digital era of communication, Electronic mail is an important, and a popular means of communication. An Email inbox is often rife with different messages ranging from High Importance to Low to spam. In order to not lose sight of important emails, it is necessary to organize the emails in proper categories. In my paper, I will be presenting an implementation approach, involving self-learning by creation of pre-built dataset using user preferences and background knowledge for Email categorization.

ACKNOWLEDGEMENTS

Firstly I would like to express my sincere gratitude to my advisor, Prof. Wei Jin for the continuous support extended towards my Masters Paper research and implementation. Her guidance and motivation helped me throughout the study and research process.

A very special thanks goes to the rest of my committee members, Prof. Salem Saeed, and Prof. Na Gong for their invaluable time.

I would like to thank my parents, my in-laws for their constant support and love.

Last but not the least, I would like to thank my husband and my daughter without whom I would not have been able to pursue my dream.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

The idea for this paper came into being a few summers back when owing to project deadlines and summer vacations; I was not accessing my emails on a frequent basis. And the day I opened my inbox, staring at me were some 250 unread messages. I had to go through each and every message to ensure that I was not deleting any important emails. Each scan made me wish to have had an automated system in place to sort my emails thus making the task less time consuming. And it was like a light bulb moment, why not work on an implementation to make use of text classification / categorization techniques to organize / manage my Emails. I was introduced to the concept of Text Classification algorithms in the Information Retrieval course, and it seemed appropriate to use my theoretical knowledge for the implementation thus strengthening the knowledge I had gained in my coursework.

There are several reasons why Email is a prevalent means of communication in today's world. Email can be sent instantaneously, it is useful for keeping records as it forms a virtual trail of whom it came from, what was the subject and the content, is a good means to market products and cheapest way for businesses to communicate. These days, it is common to have more than one email address - Personal emails, work emails, school / college emails. In general, everyone receives minimum 20 - 40 emails every day. For some maybe much more than that. It is a time consuming task to go through each and every email to ensure if the email is of any significance. Many of the emails are promotional offers from shopping outlets/magazines you have subscribed to. It has the potential of flooding the inbox and leading to an unorganized mess. Though recent progress has been made by email providers, by allowing the creation of folders for sorting/grouping emails, the process is mostly manual. Some Email providers identify Spam/Junk emails and group them in Spam/Junk folders.

In this digital age of ‘smart computing’, it is logical to have an automatic task for organizing emails - a classification algorithm to process the email and organize them into appropriate folders in accordance with user relevance.

The main objective of this paper is to implement a method to classify emails with a pre-built dataset/ dictionary with minimum user interaction. Most of the methods proposed, involve a training set, which is manually classified by the user. The main contribution of this paper is to develop a method that organizes the user’s email messages into appropriate folders / labels based on their preferences without having the users to sort manually and move the messages into folders / labels thus leading to precious time being saved, improved efficiency and better search of email messages.

The paper is organized as follows: Section (2) gives a background Text Classification and the different types of text classification methods. Section (3) describes the related relevant research ideas and how the text classification techniques were used for email classification. Section (4) outlines the proposed approach while Section (5) delves deep into the implementation of the algorithm. Section (6) shows the experimental evaluation.

2. BACKGROUND

Generally, the main tool for email management is text classification [1] [2] [3]. Different text classification algorithms can be used in classifying the email inbox.

The research of text classification algorithms has gained importance in the recent years due the enormous generation of text documents with the advent of Internet. Text categorization (or text classification) is the method of assigning documents / text to predefined categories based on their content [4]. In simple words, given a set of predefined categories / classes, the input set of documents are sorted and assigned to the appropriate category / class. For example, a newspaper is divided based on the subject category: Sports, Entertainment, Politics and so on. Text classification is often used to classify text collections into relevant categories; as in Newspaper articles and Twitter feeds. It is used to filter spam / junk emails, to improve search engine results. Two types of approaches are followed for text classification: Rule-Based and machine learning based [5]. Rule-based approach follows the methodology of defining rules manually and classifying the text documents based on these rules. On the other hand, the Machine Learning approach is an automated approach where the rules are automatically defined based on sample documents or prior knowledge.

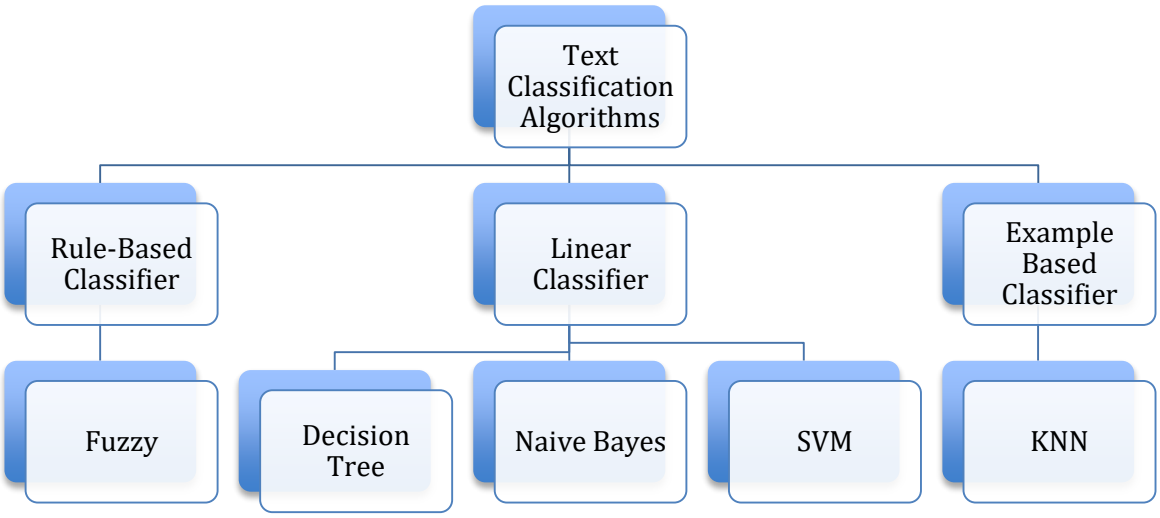

Figure 1. Taxonomy Structure of Text Classification Algorithms [15]

The above Figure (1) [15] shows the taxonomy structure for text classification algorithm as per the survey conducted by K.Saruladha and L.Sasireka [15].

2.1. Rule-Based Classifiers

The rule-based classifiers are composed of a set of If-Then rules for classification. The rules are mutually exclusive and exhaustive. They are equally easy to interpret as they are to generate. Rule-based classifiers are used across many fields like medicine, biology, spam detection, and filtering.

- **Fuzzy Rule Classifiers** – The most popular rule-based classifiers are the Fuzzy classifiers. The underlying idea is to use a set of inference rules. In other words, a set of input and output data is first converted into fuzzy sets using linguistic variables and terms. A set of fuzzy rules, which are simple If-Then rules with a condition and an inference are applied to the fuzzy sets and based on the inference of the rules a mapping is defined between the input and output data.

  Examples of Fuzzy Rules:

- IF Temperature is Below 0 THEN wear a winter jacket.
- IF Temperature is Above 40 and Below 60 THEN wear a light jacket.
- IF Sender is Unknown THEN Email is SPAM.
- IF you work hard THEN you will succeed.

2.2. Linear Classifiers

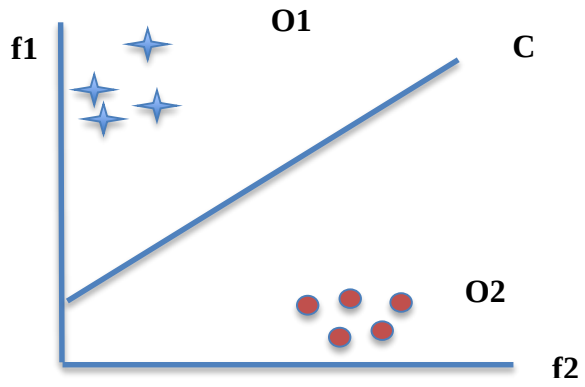

Figure 2. Linear Classifiers [18]

Linear classifiers base their classification on specific characteristics called features and a linear combination of features is taken into consideration. In Figure (2), it can be seen that there are two classes of objects O1 and O2 separated by a classifier c. It also shows two features f1 and f2. O1 shows high affinity to f1 while O2 shows high affinity to f2, in other words, O1 has more f1 features, and vice versa. Classifier c is a linear classifier as it is a linear combination of both features [18]. For each class of objects, weights are assigned to each feature based on the frequency of occurrence in each class and probability of occurrence.

Linear classifiers are used in machine learning approaches. Decision Trees, Naïve Bayes, SVM are linear classifiers.

- **Decision Trees**

Decision tree is a divide and conquer classification method made up of a tree-like graph. The graph tree consists of nodes that are attributes and the emerging branches. The branches are labeled by the weight that the attribute has in the text document, and the leaflets are labeled by categories [5]. It employs a top down approach starting at the root node, applying a set of rules, selecting the branch with the closest outcome and continuing the path of branch selection based on decision / outcome till it reaches a leaf that represents the end resulting category. Decision Trees are widely used in operations research, Computational Biology, and Bioinformatics.

- **Naive Bayes**

Naïve Bayes belongs to the probabilistic classifier family of Bayesian approaches based on the Bayes theorem with independent assumptions. Independent assumptions mean Naïve Bayes follows the assumption that the presence or absence of one feature of a class is not related to the absence or presence of another class feature. Even if the different features are interdependent, the Naïve Bayes classifier takes into account that all of the features / properties contribute in their own way in the process of classification.

- **Support Vector Machines**

Support Vector Machines is a model, which employs the supervised learning algorithm. It is used to distinguish two groups and classify new entries in the correct groups. Given a set of labeled training samples, the support vector machine model outputs an optimal hyper plane that classifies the set and assigns them to the appropriate group. It is used in various applications used in medical science such as protein classification, gene expression data classification, outlier's detection and many others.

2.3. Example-Based Classifier

Example-based classifiers classify a new document with the help of a training data. It starts by finding the K^{th} nearest neighbor of the new document in the training data and based on majority voting, and the new document is assigned to a specific category. KNN or K-Nearest Neighbor is the most popular example-based classifier.

- **K-Nearest Neighbor**

The foundation of classifying documents in K-Nearest Neighbor classification method is based on a similarity measure like the distance measure. K-Nearest Neighbor is a classification algorithm where objects are classified by taking a set of labeled training examples and voting them based on the minimum or smallest distance from each object [5]. This data mining method is used in different fields of Text mining, agriculture patterns, stock market forecasting, medical field in the patient analysis.

3. RELATED WORKS

Recently my 8-year old daughter came to me with a question related to her school assignment on letter writing. She asked me an innocent question, “Mom, I have never seen you write letters. Is that not something you do with emails?” The question made me ponder as to the last time I wrote a letter and made me realize the impact Email has had over our lives. Electronic mail, also popularly known as email or e-mail, was invented way back in the early 1970’s. Email is similar to the concept of a traditional postal service, the main constituent for the email message is the recipient address but it scores over the "paper mail" on two fronts: the speed at which the email is delivered (literally in a flash) to one or several people and at a lower cost [20].

In today’s age, technology has changed the way people communicate in astronomical ways. Everything is digitized, from sending communication about company promotions / marketing to tweeting about a new addition in the family. Social media like Twitter; Facebook have taken over the digital world of communication. In spite of the popularity of these social media applications, Electronic mail still stands out as the most preferred mode of communication. The number of user accounts an Email has, is nearly three times more than Facebook and Twitter [21].

A sheer volume of emails received and sent often inundates an average email user. More than 90% of emails sent are usually junk or spam email. The user is faced with a task of spending a considerable amount of time sifting through the emails and deleting if unwanted, manually organizing them, which involves first creating the folders and then moving the messages to the created folders. The manual intervention needed, the amount of time spent and the crucial role of electronic mail in everyday life have led to the importance given to the research on the classification of emails. Some of the main focuses in this area are:

1. Spam Filtering – Spam email also known, as Unsolicited Bulk Email is a matter of concern to email users, as they always seem to come in bulk. Spam email not only mounts to waste of time but also takes up a lot of storage space. Classifying emails into spam and non-spam emails have been a major topic of research.
2. Email Classification using text categorization algorithms such as nearest neighbor, Bayesian, Maximum Entropy, Ripper.
3. Automatic categorization of email into folders.
4. Classification based on a graphical representation.
5. Feature extraction involving extracting features/ patterns from the email structure (body, sender, and subject) to classify emails into appropriate groups.

3.1. Spam and Non-spam Email Classification

Spam emails also known as Unsolicited Commercial Email or UCE have been part of the Email world from a relatively early age from its inception. The first spam email was sent in May of 1978, but it was identified as a problem in 1982. With the growth and popularity of Internet, the volume of spam emails has increased too. As per reports in PC magazine in 2009, nearly 98% emails were spam emails. Spam emails generally promote Internet-based sales, but might also promote telesales or other types of sales.

An email can be categorized as spam based on following components:

- Manually marked as ‘Spam’ by the user in earlier instances.
- From an unsafe / blacklisted domain.
- Unknown Sender.
- Malicious links.
- The presence of unknown Language.

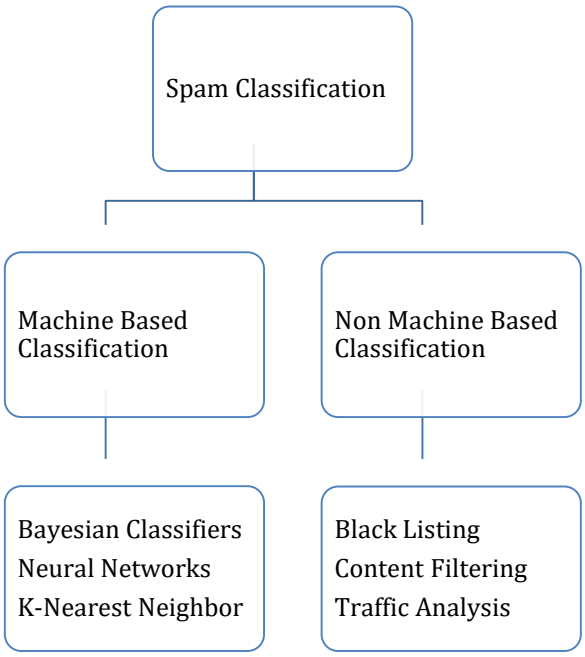

Figure 3. Spam Classification Categories [19]

As shown in figure (3) spam filtering or classification is divided into two categories

- Machine based
- Non Machine based

➢ **Machine based classification**

McLeod et.al [16], proposed an adaptive ontology approach for spam email filtering. The ontology filter was constantly evolving based on the user's preferences making it more adaptable.

Sahami et.al [3], proposed the use of domain specific features like the sender types with the Bayesian classifier for junk email filtering. Phrases like "Free Money", over embellished punctuation marks "!!!!" were considered as domain specific features. By considering these additional features along with the natural Email message content, an improvement was found in the accuracy of filters.

- **Non Machine based classification**

In the early days, when spam had just emerged, the anti-spam techniques belonged to the non-machine based genre. The techniques varied from maintaining a list of unsafe keywords, a blacklist of unsafe domains or spammers.

- Content Filtering – It is a method of maintaining a list of words or topics of which one is sure never to receive emails about and filtering the emails by these words achieved the purpose of filtering spam emails.
- Black Listing – Create a list of IP addresses of unsafe domains / hosts / spammers. The email messages can be filtered as per the list. This filter is considered fast and simple but has the disadvantage that it is easy to breakthrough for the spammer by imitating a sender's email address.

3.2. General Email Classification

Wang et.al [6] have proposed a half supervised learning method involving processing of feedback from the user. The method applies different classification rules to each section of the email – email subject; body, from and to addresses, the characteristics that are always read by the user. Based on the classification results, the emails are assigned to the correct group.

Bekeerman et.al [7] & Boryczka et.al [8] give an insight into the automatic categorization of emails into folders. Bekeerman et.al [7] uses the time incremental split theory by having a training set based on the first half of messages and then testing the training sample on the second half. It used three different classification algorithms - Maximum Entropy, Naive Bayes, and Support Vector Machine (SVM) and presented a newer version of the Wide-margin Winnow algorithm. Boryczka et.al [8] has proposed a new approach based on the Ant Colony Optimization for the classification of email messages.

Saxena et.al [9] proposed an approach based on Ant clustering for email categorization. It uses a phases approach with first training phase involving the manual sorting of emails into folders by the users. It is followed by the testing phase where the emails with already determined categories are tested and the last document-processing phase in which the Ant clustering algorithm is applied to emails whose categories need to be determined.

The aim of Arey et.al [10] is to automate the process of email classification. The approach is based on the hypothesis that the structure and patterns can be extracted and used in classifying the incoming emails. It uses a graphical representation of the email structure (header, body) and the relationship between the various terms occurring in the structure.

Vira et.al [11] has proposed an algorithm that uses Bayesian Theorem to classify emails. The conditional probability is used on email's textual content using keywords from manually classified emails by the user.

Cui et.al [17] proposed an Email classification method based on Neural Networks. This method was used to classify personal emails that were considered as plain text and made use of Personal Component Analysis (PCA) as a preprocessor to Neural Networks thus resulting in the reduction of data making the classification process easier.

3.3. Text Classification Techniques Implemented for Email Domain

In the introduction, I briefly described the text classification methods. This section describes how some of those text classification methods are used to classify emails and the different software tools used for Email classification.

The content that makes up an email body is either textual or non-textual or a mixture of both. Current email services allow the use of both plain text and HTML. It is short and concise as compared to text documents. The content may vary from website links, images, and attachments, personal to social and promotional. In some of the research done, an email pre-processing step included of ripping off the non-textual data.

3.3.1. Spam Classification Using Decision Trees

Decision Trees are made up of two nodes – decision (parent) nodes and leaf nodes. Rules are decided and applied to a training set. The best rule is applied to the parent node and splitting occurs resulting in the leaf nodes. Rules are applied recursively till the achieved result, or the leaf nodes cannot be further split. For email classification, as seen in many proposed approaches, a training set of emails is created. Applying recursive rules to the root of the tree that is the sample training set creates a decision tree. Rules / attributes are applied to the root. Division / splitting occurs based on the selected attributes.

Figure (4) shows a decision tree construction for spam email classification.

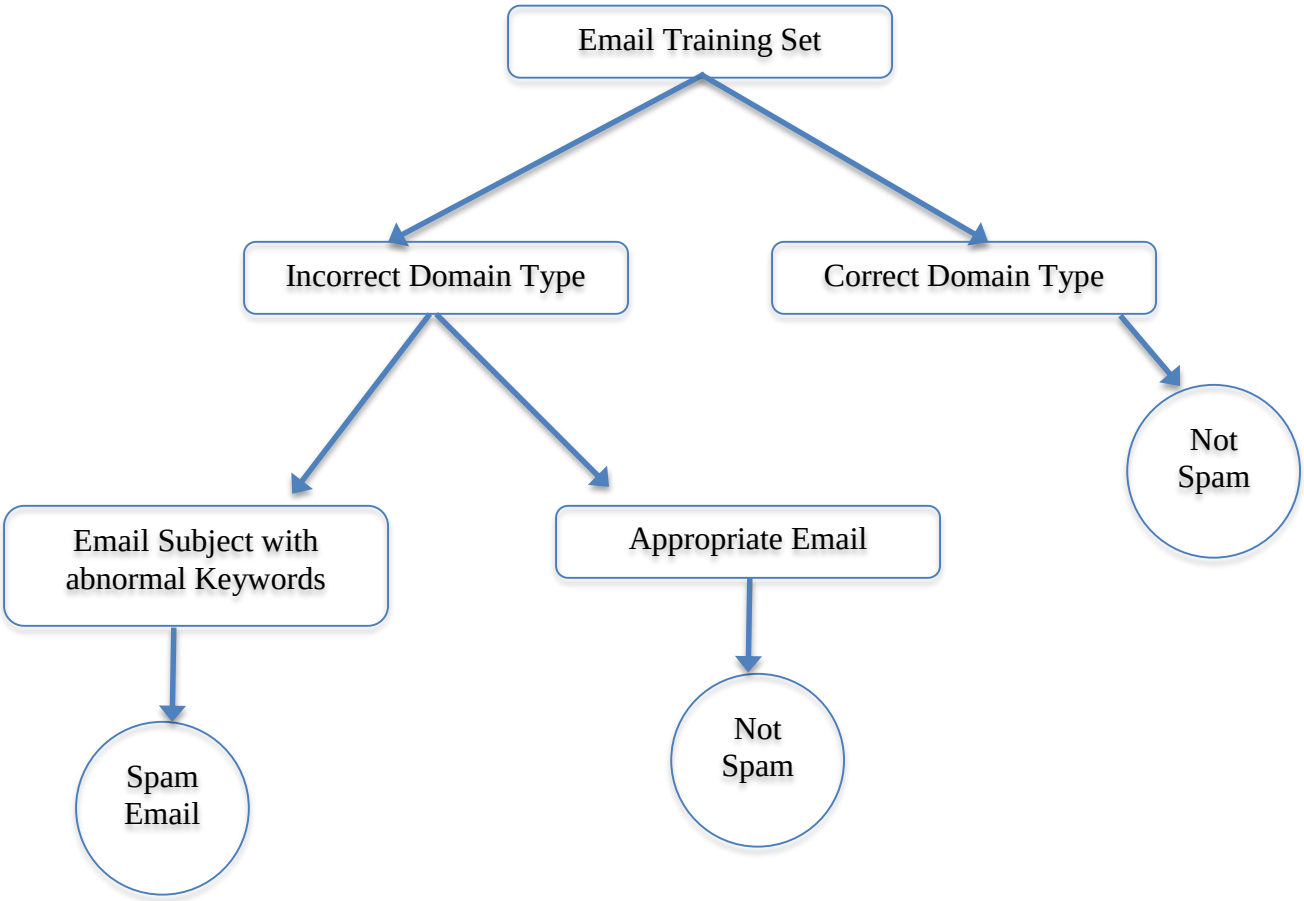

Figure 4. Decision Tree Implementation for Email Classification

In the above decision tree, a training email sample is at the root. Rules are applied to different email attributes. The domain type is the first attribute checked to see if the email is from a valid domain. If yes, it is classified as a valid email. If not, further rules are applied. The next attribute is the email subject. If it contains abnormal words, it is classified as a Spam Email. The decision tree shown is only a case of partial classification applied. The decision rules can be further applied to email sender and subject till no splitting can be done.

The advantage of Decision Tree is that it is very simple to build and easy to understand, interpret and evaluate. The main disadvantage of this classification technique is that a small change in the input can lead to large changes.

3.3.2. Email Classification Using Naïve Bayes Classifier

A Naïve Bayes classifier belongs to the Bayes probabilistic classifier family. It is widely used for spam detection in emails. It uses the conditional probability of Bayes theorem to analyze each attribute individually and independent of each other. The below spam conditional probability rule was specified by Vira et.al [11]

P (S|E (e1, e2...en)) = P (E (e1, e2...en)| S)*P(S) / P (E (e1, e2…en)… Eq. (1) [11]

Where, P (S|E) is the probability that an email E is spam S.

The algorithm specified in [11] consists of 2 phases. Email classification was done using Bayesian Theorem. The learning phase constitutes the first phase, which involved a creation of training set of emails. Two categories were pre-defined – Work and Personal. The users had to specify manually which emails belonged to the work or personal category. A database was created to store the keywords associated with the training set and categories. The classification was done by preprocessing the email, and Bayesian Theorem was applied to compare the email contents with the keyword database and to determine the probability of which categories the emails belonged to.

3.3.3. K-Nearest Neighbor Email Classification Method

K-Nearest Neighbor classifier is used for filtering out spam emails. A training example set t, of email messages, is created. The main idea is, given an email message m; determine the k^{th} nearest neighbor of m from the training example t. To make the determination, the distance between the messages is calculated. Euclidean distance is the most common metric used. If

among the k nearest neighbors of m, x or more are spam messages than m is categorized as a spam mail else it is added to the legit email category.

3.3.4. Email Classification Using TF-IDF Classifiers

TF-IDF classifiers are the most commonly used and popular classifiers used for the classification of emails. Different types of classifiers were proposed for the same. Segal and Kephart proposed a TF-IDF classifier in [13] which suggested the top n categories an unclassified email belonged too. This was determined based on the TF-IDF principle for calculating the weights of the word frequency vector.

Another classifier was proposed by Cohen (1996) in which, an email is represented as a weighted vector [2]. TF-IDF weight of the terms was calculated. A threshold was set, and a newly incoming email was classified into a specific category if the similarity score (resulting product of email and category vector) was less than the threshold.

3.3.5. Software Tools for Email Classification

MailCat is an intelligent assistant tool developed by Segal and Kephart to help users organize their emails into different folders [13]. The main idea behind the tool is to learn from the user tendency or habits when accessing emails. Based on what is learned, the tool can predict the top three folders or categories a user is likely to select for an email message. For user convenience, the tool provides 3 shortcut buttons for moving the messages to these folders. The prediction accuracy was between 60-90%. This tool was developed mainly for emails but can also be used for bookmarks, files, and other text documents.

IFile is an email-filtering tool developed by Jason Rennie [14]. It is based on the Naïve Bayes classification algorithm and consists of 3 layers. A C executable, which is tasked with

storing and maintaining the classification model and generate the class labels for emails. The wrapper script layers filter the incoming emails and update the classification model. And the last Tcl code is used as the lookup into the user interface for the IFile to be used with minimum user interference.

4. PROPOSED APPROACH

Going by personal experience, and information from colleagues and friends, I have come to the conclusion that an average person receives anywhere from 20 – 200 emails every day. In such a scenario, it is quite natural to miss an important email, which can sometimes lead to serious issues. This brings in to picture the need for an effective tool / process to manage and classify emails automatically thus saving time.

The email account is personal to a user that can be used as a personal email or a professional email account. When a user receives an email, the first thing the user checks is who it is from. If the user approves of the sender, he opens the email and reads it. If not, he deletes the email or keeps it for later read. If the sender is unknown, he checks the email subject and then opens the email. The email he receives is based on his / her personal / professional preferences. Taking into account the important role a user preference plays into the email messages received, my approach revolves around inculcating the knowledge of a user's preferences in the email classification process. The approach proposed in this paper is the use of pre-defined knowledge of user's preferences and application of classification rules to the different sections of Email based on the knowledge acquired. It will also involve self-learning based on the additional information from the classified emails. Before outlining the paper approach, it is important to walk through the email structure. The next sub section will talk in brief about the different elements that constitute an email message.

4.1. Email Structure

Currently, there are more 100-email service providers and more than 3.1 billion email accounts worldwide. Some of the popular email providers are Gmail, Outlook, Hotmail, and Yahoo. This paper will be using Gmail API for the email message processing. Below Figure (5) shows a screenshot of how a Gmail account looks like.

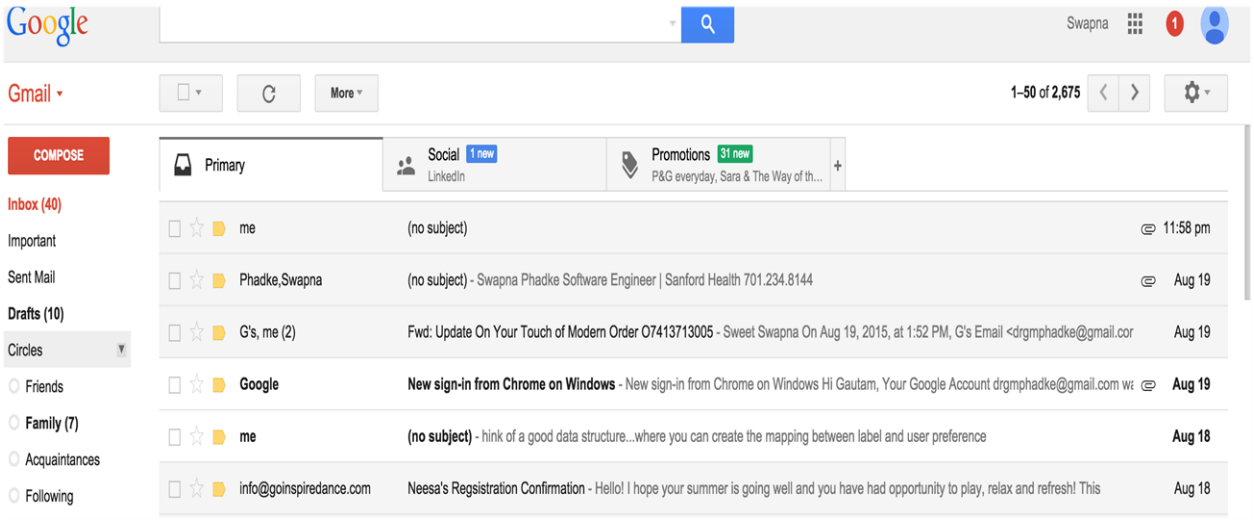

Figure 5. Email Structure

Email message mainly constitutes of 2 parts:

- Email Headers
- Email Body

4.1.1. Email Headers

Email Header is composed of the fields that summarize the email lifecycle from the information of the sender, the recipient, the timestamp an email was sent or received, the subject of the email. The Primary fields are:

From (Email Sender): This is the email address of the person who sent the email message. If a person sends an email, the 'from' field will be his own email address whereas if he receives an email message, the 'from' field will be from the person who sent the email.

To (Email Recipient): This is the email address of the person the email is addressed to.

Date-Time: This field specifies the date and time the email message was sent.

Subject: This field specifies the subject of the email message, which gives a brief idea about what the message content might be.

CC (Carbon Copy): This field specifies that an email message can be sent to more than one person by comma separating the email address.

BCC (Blind Carbon Copy): This field is similar to the above CC field. The only difference is that the recipient of the email message will not be able to see the email addresses of other people the email message was sent to.

Message ID: This is an important part of the email message that is hidden from the common user view. This field is the unique identifier of a message and can be used when the email message has a thread of emails, and the message has been replied to.

Secondary Fields are:

Content-Type: Specifies the text and character format of the message.

Importance: Specifies if the email is of urgent or important nature.

In-Reply-To: Specifies if the email was replied to.

X-Originating-IP: It tells us the IP address of the sender.

Content-Disposition: Allows attachments.

Content-Encoding: It specifies if binary data is portrayed as ASCII text.

MIME format: Specifies the MIME format used.

4.1.2. Email Body

Email body is the textual content, which forms the content of the message. The content can include text, images or unstructured data. Attachments also can be sent or received as part of the email body. It optionally contains a signature at the end of the message.

Only 7-bit US ASCII characters are supported in email messages where each line is not more than 76 characters and ends with CRLF (\r\n) [24].

4.2. Outline of Proposed Method

Most Email providers allow users to sort their messages into folders. Theoretically speaking, one might think that the task of sorting a message is trivial or inconsequential. However, in practice, many find the task of deciding which folders / labels to create and then determining which email messages need to be assigned to which label / folder a mentally straining exercise not to forget the time spent. The intention of the method proposed is to take away this load from the user. The user involvement in this method is only to fill a web form at the first stage, and the learning process takes over after that.

The proposed method consists of 4 steps:

1. **Creation of User preference dataset -** User will be provided with a web form to fill. The form will consist of a set of questions, which the user needs to answer. These answers will form the basis of the user preferences dataset, which will be used with the classification rules.
2. **Training Sample Set creation -** A set of emails ranging from personal to promotional is grouped together in the training set.

3. **A Key value pair Dictionary creation** - This dictionary will store the most frequent words associated with each key found in the email, subject, and the email body.
4. **Use of Classification Rules** - This step will be the last and determining step where different rules will be generated and applied to the different Email sections like header, subject, and body.

Figure (6) shows the process flow:

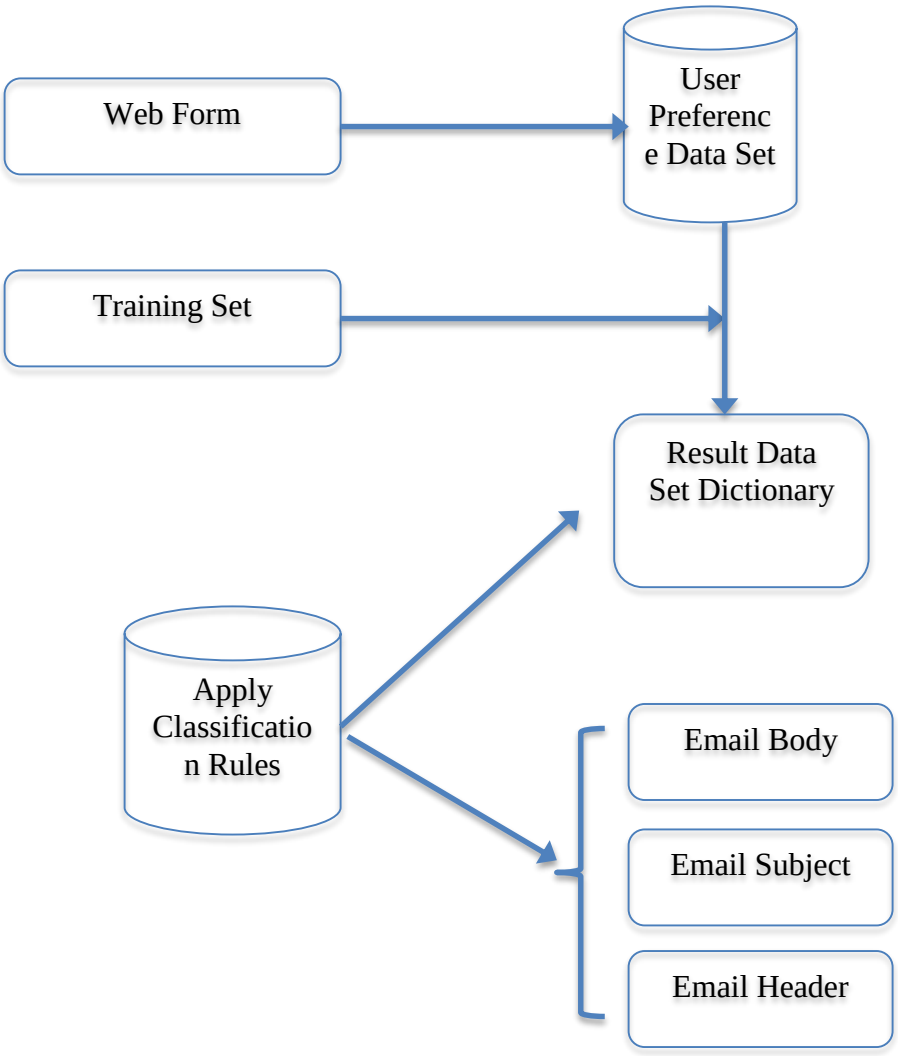

Figure 6. Proposed Approach Process Flow

4.3. Pseudocode for Classification Using Self Learning

1. User Preference Dataset U = {Name, School Name, College Name, Parenting, Shopping Domain, Social Media}.
2. Training sample T = {$email_1$, $email_2$…$email_n$}.
3. Tokenize all the messages in T into separate tokens t.
4. For Each item for the Key in U

    For Each email message E in T

        Build Keyword dictionary K.

        Determine most frequent words from Email Body.

        If (E contains item) Then

            Store the Key as the Key and item + most frequent words as values

    End If

    End For

End For

5. Retrieve Email Messages
6. For I = 1 to Total No. of Messages
    a. Tokenize Email from, Email Subject and Email Body.
    b. For Each Key in K

        For (J = 1 to Length (Email from))

        If (KeywordDictionary.Values Contains Email from (J)) Then

            Add Email Sender Address and title to K.

            Classify Email to the folder name corresponding to the key.

        End If

```
                End For on J
                If no match found in Email from Then
                    For (L = 1 to Length (Email Subject))
                If (KeywordDictionary.Values Contains Email Subject (L)) Then
                    Add Email Sender Address and title to K.
                    Classify Email to the folder name corresponding to the key.
                End If
                    End For on L
            End If
                If no match found in Email Subject Then
                    For (M = 1 to Length (Email Body))
                If (KeywordDictionary.Values Contains Email Body (M)) Then
                    Add Email Sender Address and title to K.
                    Classify Email to the folder name corresponding to the key.
                    Store most frequent words from Email Body to K.
                End If
                    End For on M
            End If
        End For Each Loop
End For on I
```

5. IMPLEMENTATION

This section describes the work flow of the proposed approach and walks through the implementation process.

5.1. Environment Used for Implementation

The email classification approach proposed in this paper is developed as a web application. I chose ASP.NET C#, as it is the programming language I have started using at my work and I believed that working on this paper will help me strengthen my basics and add to my existing knowledge giving me an opportunity to learn different aspects of the language. Below are the software/hardware entities used to build the classification application:-

- **IDE Tool –** Visual Studio 2013 is the new edition of Visual Studio, which allows to create apps in one unified Integrated Development Environment (IDE) [22]. It supports different built-in languages like C, C++, VB.NET, C#.
- **Application Framework** – The web application framework used in the implementation is ASP.NET MVC. ASP.NET MVC, developed by Microsoft is based on the model-view-controller (MVC) pattern [22]. A software application built as a result of using the framework is a collection of 3 layers:

  Model (Application Core layer), View (Data Display layer), and Controller (input handling layer).
- **Programming Language** – The programming language used in the implementation is C#. C# is an object-oriented language designed for Microsoft's .NET framework.
- **Gmail API** – Gmail API is a restful API that was used to create a security token for authorized access to read and modify Gmail messages.

5.2. Phase 1 - Creation of Data Collection Set

This phase is the building block or foundation of the proposed method. It will involve:

a. Creation of a dataset consisting of user preferences.

b. Collect Email messages to form a training set.

c. Creation of Key-Value pair Keywords Dictionary.

5.2.1. User Preference Dataset Creation

As specified earlier, the main purpose of the proposed method is to minimize the manual user involvement thus freeing the user of the additional load of creating folders and sorting the emails. The user interaction is limited to filling a questionnaire. The starting point was to create a web form for the user to access. The web form is a set of questions related to the general information pertaining the user's interests and background. The questions range from personal questions like gender, education to professional like company worked / working at, professional background. The user will be required to fill the form. The answers provided by the users play an important role in the creation of the Keyword dictionary. On submission of the web form, the data is stored in a JSON object.

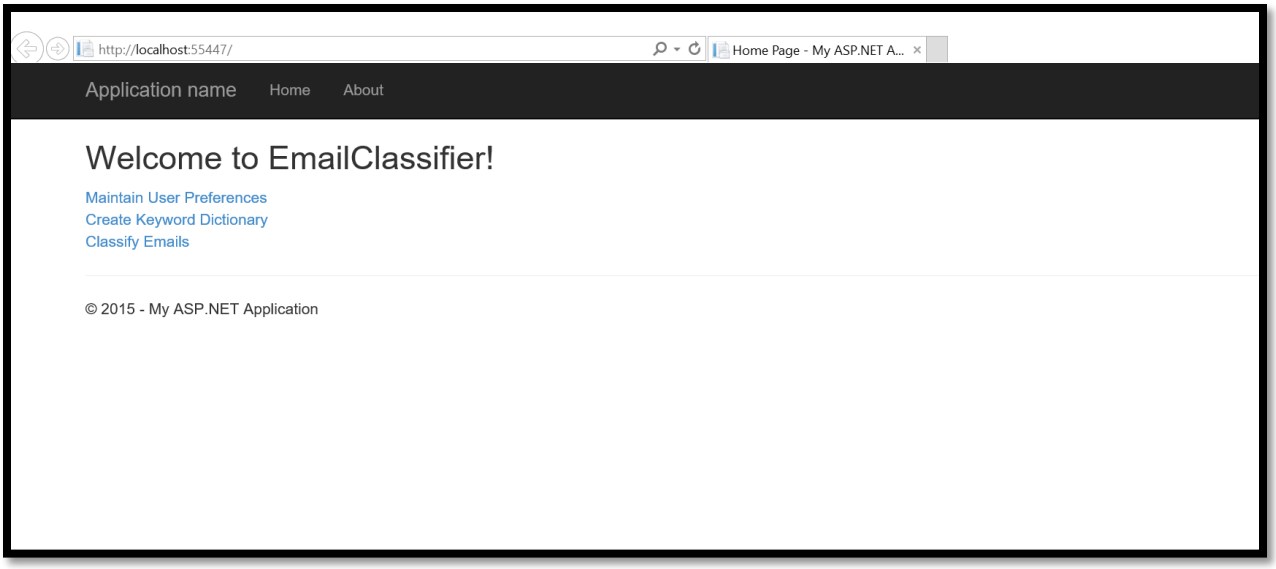

Figure 7. Email Classification Application Home Page

Above figure (7) shows the Home page of the application. Three links are available on the home page.

1. Maintain User Preferences – This link opens up the view page where the users can add/ edit / update their preferences.
2. Create Keyword Dictionary – Clicking on this link, starts a background process, which creates the Keyword dictionary with user preferences submitted by the user.
3. Classify Emails – This link involves the scanning of emails, folder creation, and assigning the emails into the folders.

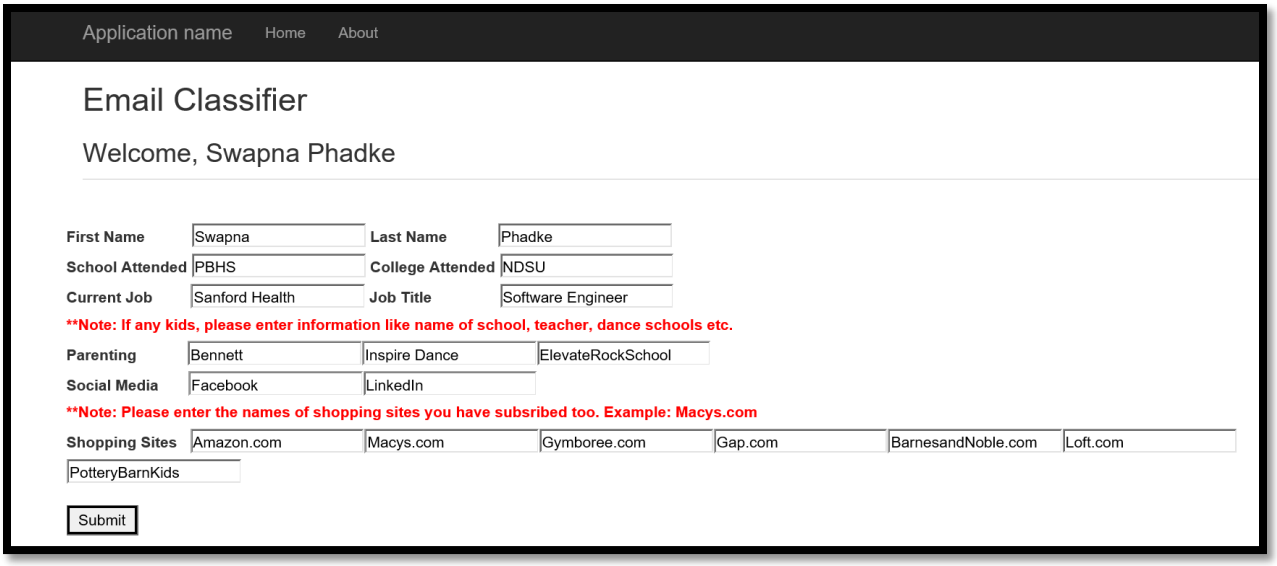

Figure 8. Entry Form for User

Figure (8) shows the entry form webpage for the user. The user has to enter information related to the school, colleges he attended, shopping sites he has subscribed to. If any kids, then the related information like kids schools. Once the user submits the information, the data is stored as a JSON data object. This data object forms the basis of classification as it helps to understand the personal preferences of the user. It also forms the basis for folder creation. The keys in the JSON object that are the labels used in the web form are used as the source name for the folders. For example, Parenting, SocialMedia are some of the folders that will be created in the classification phase.

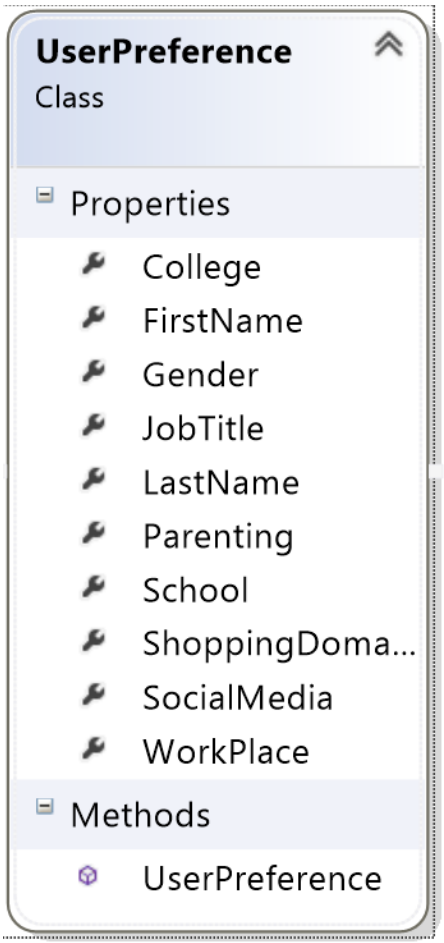

Figure 9. Class Diagram for UserPreference Data Object

The JSON object stored as a result of the web form submission is

```
{
    "FirstName"      : "Swapna",
    "LastName"       : "Phadke",
    "Gender"         : "Female",
    "School"         : "PBHS",
    "College"        : "NDSU",
    "WorkPlace"      : "Sanford Health",
    "JobTitle"       : "Software Engineer",
    "Parenting"      : ["Bennett Elementary","Inspire Dance", "Elevate Rock School"],
    "SocialMedia"    : ["Facebook","LinkedIn"],
    "ShoppingDomain" :   ["Amazon.com","Macys.com","Gymboree.com",
"Gap.com","BarnesandNoble.com","Loft.com”,"PotteryBarnKids"]}
```

Figure 10. User Preferences Data Object

5.2.2. Training Set

The training set for this paper is formed from a collection of emails selected from my personal Gmail Account. The reason for selection of Gmail provider service was the access permissions provided by the Gmail API. Gmail API is a REST service, which is used to access the Gmail messages and analyze them. It allows for authorized access to the Gmail mailbox and hence was appropriate for the implementation of the proposed method. The idea behind the selection of emails in the training set was to include emails with the different type of content, ranging from personal to professional. The training set consists of: -

1. Emails belonging to promotional type for example – emails from shopping site domains like Macys, Crate and Barrel.
2. Emails belonging to parenting type – emails related to my daughter.
3. Emails falling under the Social Media Category – emails received from Facebook, LinkedIn.
4. Emails belonging to professional type – emails received from my work place.

Examples of Emails in the training set.

```
[
    {
    "From": "Gap@email.gap.com",
    "Subject": "40% off + FREE 2-day shipping",
    "Body": "Do not miss the Sale. Mon - Wed."
    },


    {
    "From": "PotteryBarnKids@email.PotteryBarnKids.com",
    "Subject": "Last Day Extra 20% off",
    "Body": "Online Sale. Only for the weekend. Hurry. Online Only"
    },


    {
    "From": "bennettptavolunteers@gmail.com",
    "Subject": "Volunteers needed for first day of school",
    "Body": "Hello all and welcome back. We need volunteers for kindergarten pool,
to pt labels on 1st grade backpacks. If you are able to help out please let us know.
Thank you in advance."
    },

    {
    "From": "davidr@radianthomes.com",
    "Subject": "planters",
    "Body": "Hi Swapna and Gautam The cost of the cedar planters is $150. This includes
both boxes."
    },

    {
    "From": "mikes@sanfordhealth.org",
    "Subject": "Assignment Review Request",
    "Body": "Please review your assignment on ticket 0287865. Please contact me if you
have any questions. Thank you. Mike S"
    }
]
```

Figure 11. Training Set Email Examples

5.2.3. Keyword Dictionary Creation

Keyword dictionary is the resulting dataset of applying classification rules to the training set and the user preferences dataset. The keyword dictionary used is stored in the key-value pair format. A key-value pair consists of two interlinked data items; a key, which is a distinct identifier of a specific item and its associated values. The key value pair format used for the keyword dictionary consists of,

- Keys that specify the label or category an email will be classified into.
- The value represents the most frequency words that appeared in a training set email for the corresponding user preference dataset entry.

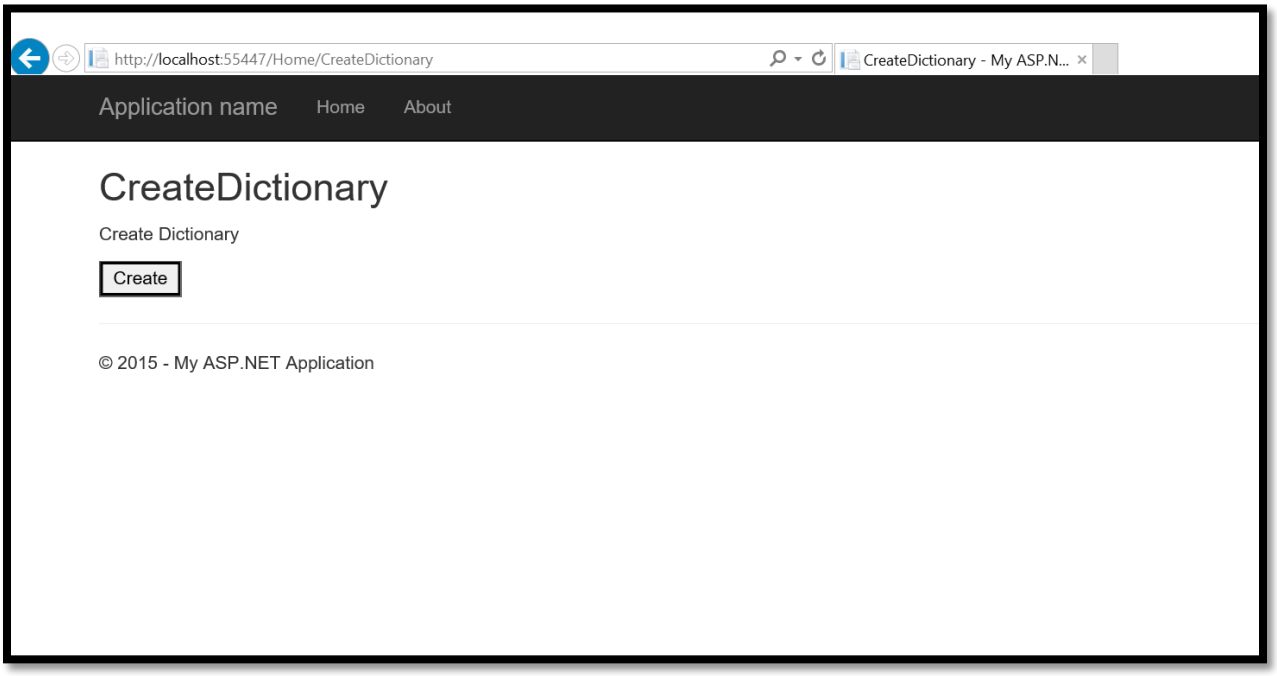

Figure 12. Webpage for Keyword Dictionary Creation

Keyword Dictionary object is of the form as shown below. It is a combination of keywords from the user preference data object and the training set. The words in bold are the most frequency matching words from the training set.

The Keyword Dictionary JSON object is shown as in the format below.

```
{
   "ShoppingDomain": [ "Amazon.com",  "Macys.com", "Gymboree.com",
      "Gap.com", "BarnesandNoble.com", "Loft.com",
       "Last", "Extra", "20%", "Online", "Sale.", "Only", "weekend.", "Hurry",
"PotteryBarnKids" ],

   "SocialMedia" : [ "Facebook", "LinkedIn" ],

   "Parenting"      : [ "Bennett Elementary", "Inspire Dance", "ElevateRockSchool" ],

   "Appointment"     : [ "appointment", "reminder"  ],

   "Professional" : [ "Sanford Health", "Software Engineer" ]

   "Travel"          :["Itinerary","reservation","flight","E-Ticket","Airlines"]

  }
```

Figure 13. Keyword Dictionary Data Object

At this stage, the labels of folders are also created in the Gmail Inbox. The folder labels correspond to the key values in the Keyword Dictionary. For example, ShoppingDomain, Parenting, Travel are some of the folder labels created along with the keyword dictionary creation. The folder labels created can be seen in Figure (10). Thus, the user is saved from two tasks (1) thinking about an appropriate folder label and (2) creating the folder labels.

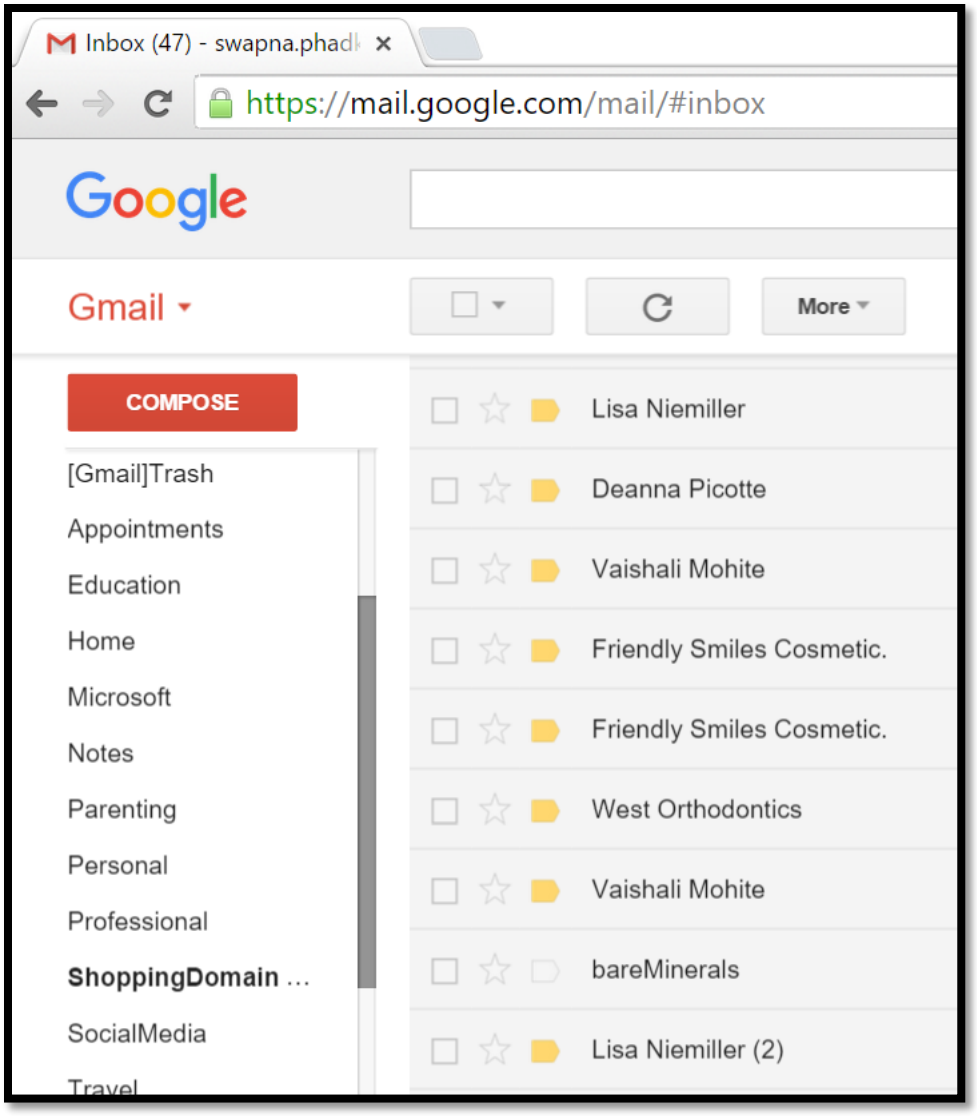

Figure 14. Email Inbox with Folders / Labels Created

5.3. Phase 2 - Rules for Classification and Self Learning

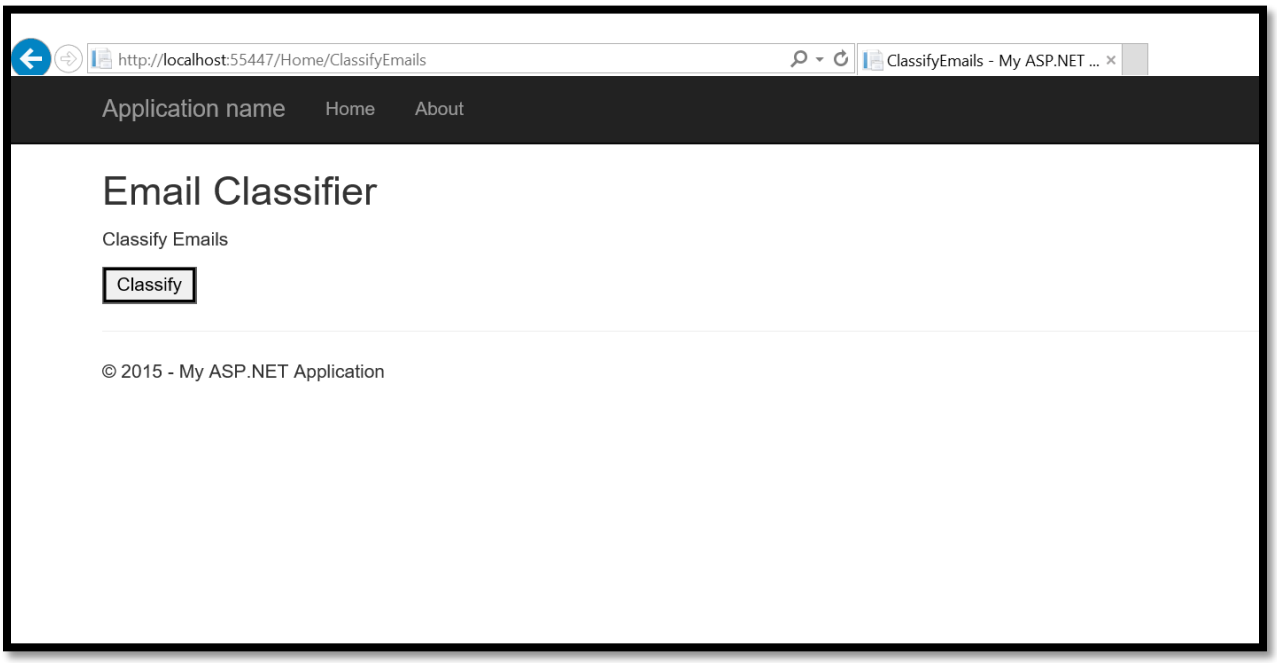

Figure 15. Webpage to Classify Emails

Once the Keyword Dictionary is ready, the classification rules of If-Then are applied to the email structure. Before initiating the classification process, the email message needs to undergo a structural conversion. Different parts of the email message, ‘from’, ‘subject’ and the ‘body parts’ are tokenized. Tokenization is a process of segmenting text into a separate string of words or characters named tokens. These tokens then will form the input for parsing.

A list of stop words is created. Stop words are the most common words in the English Language, which are filtered out to speed up the search process. A stop word list is considered in this approach as I did not want to add common words like ‘to’, ‘at’, ‘on’, ‘there’ to the Keyword dictionary as they do not provide any significant knowledge about the email message. Thus, not only it speeds up the comparison process of the email token words with the keyword dictionary but also refrains from adding nonconsequential words to the keyword dictionary.

Processing of Email 'From': The tokenized words from the Email from are compared with the keyword dictionary. If the tokens do not belong to the stop word list, it is compared to the values of the dictionary for each key – from shopping domain to personal. Once a match is found, the email message is assigned to the label of the key, for which the keyword from the dictionary matched with the 'from' tokens.

When a match is found, the tokens in the 'from' string are added to the keyword dictionary. This added keywords form a basis for self-learning as when another email from this sender is encountered, it will be assigned to the folder label by just looking at the keywords from the email 'from'.

Processing of Email Subject: The subject of an email is a summarized information of the contents of an email body. It plays an important role in an email being read or not. If the user finds the information provided in the subject line to be of any importance, he opens the message to read it or just deletes it without reading it or keeps it for future reading.

Email subject, also goes through the same processing as Email from. The tokenized words from the email subject string are compared with the keyword dictionary. Stop words are ignored. When a match is found, the email message is assigned to the label of the key, for which the keyword from the dictionary matched with the 'subject tokens.

The words in the subject are usually precise and convey the message in simple and few words. The keywords in the subject play a significant part in classification. So I have considered few of the keywords in the subject for classifying emails into a specific category. Keywords like appointments are added to the dictionary for classifying the email messages to Appointment folders. Email messages like doctor's appointment, hair cut appointment or a teacher meeting appointment will be classified to appointments category.

Keywords like ‘Itinerary’, ‘E-Ticket’ usually convey information regarding any travel or trip to be undertaken. Such emails are classified into the Travel category.

Processing of Email Body: Email body describes what the message is about. A message body may consist of text, text and images, only images, video, audio or attachments. The message undergoes the process of tokenization. All stop words are ignored. Punctuation marks are ignored.

Every token in the message are compared against each and every value of all the keys present in the dictionary. If a match is found, the tokens with a frequency value of 4 is added to the dictionary.

Classification Processing: Email From, Subject and Body tokens are scanned to find a match from the keyword dictionary. If a match is found, corresponding maximum frequency words are added to the dictionary for the key for which the match was found. If no match is found for all the keys, then the algorithm checks for the replied-to attribute of the email. If the email message has a thread of emails, i.e. the email message has been replied to by the user, then such emails are assigned to the ‘Personal’ folder. If the email message does not pass any criteria, they remain as unclassified.

Self -Learning: The approach described in this paper, also uses self- learning along with user preferences. The learning process begins when a match is found in an email token and the keyword dictionary. The email is classified into the key label. The most frequency words from the ‘from’ and body attributes are stored into the Keyword dictionary for future reference. When an email is encountered, which is not a direct match in the user preference list, the keywords added as a part of the self-learning approach are used to find a match.

For example, the user preference list used in the implementation had a user preference "Elevate rock school" for Parenting. During the experiment, an email from Lisa was found which had a match for Elevate Rock School. For the self-learning, keywords from the email like the sender email address were added to the keyword dictionary. When a second email from the same sender was encountered, the check did not find any match for the initial dictionary keywords (as this email did not have a mention of elevate rock school), but the new keywords added had a match in the sender address and thus the email was classified into the Parenting folder.

In another case, when emails from a shopping site 'Justice' were classified, the first two emails remained unclassified as no match was found between the email tokens and the keyword dictionary. But a match was found for the 3rd email and the email message was assigned to the ShoppingDomain category and the keyword dictionary was updated with the most frequent words in the message and the sender address. After this assignment and the update to the keyword dictionary, when the first 2 emails again underwent the classification process, they were assigned to the ShoppingDomain folder based on the learned keywords from the keyword dictionary.

Classification Addendums: After few rounds of classification, I did some proof reading of the classified emails to verify if the emails were classified correctly. The erroneously classified emails were analyzed. After the analysis, the classification rules were revisited to see if any improvements could be made to reduce the number of misclassified emails.

Some of the improvements done included the way the email 'from' field attributes were added to the keyword dictionary during the self-learning process. The tokenization process was done on white spaces and the character literal '@'. This caused two words that had more meaning when considered together, to be separated as different words and the match had to be

done on these words as two different entities rather than one entity. This led to erroneous classification of an email message. Also for better classification, the entire email address of the sender was added to the dictionary as opposed to the broken down address. For example, “email.xyz@gmail.com” was added as one token instead of four separate tokens in the form of email, xyz, Gmail, and com. This helped in finding a match based on a more meaningful feature than random words.

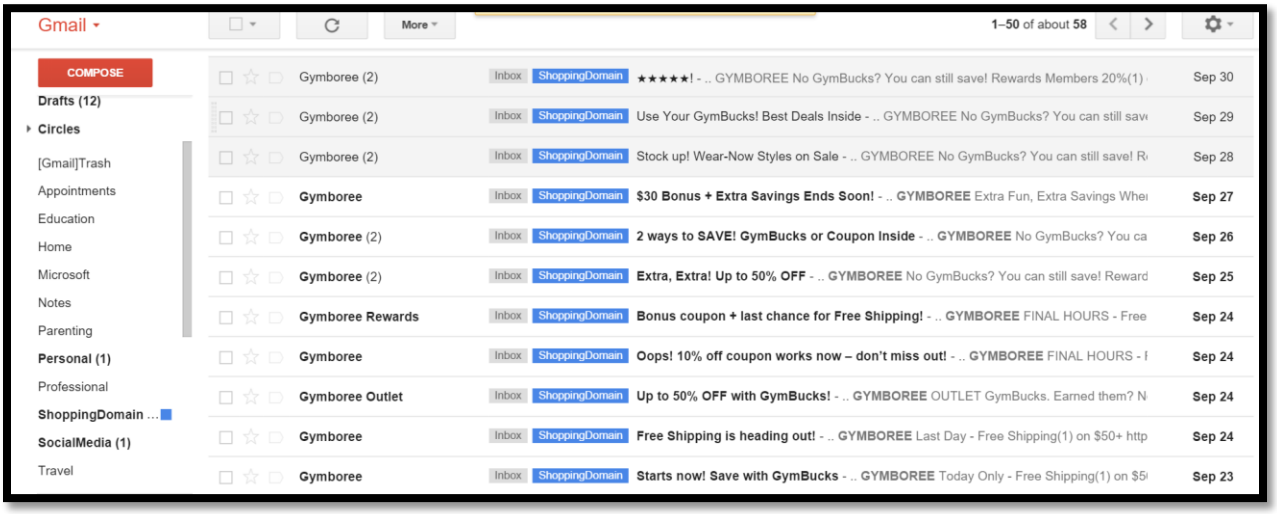

Figure 16. Email Messages Classified in ShoppingDomain Folder

Figure 17. Email Messages Classified into Two Folders

```
public EmailMessageModel BuildEmailMessage(string id)
        {
            EmailMessageModel emailmsg = new EmailMessageModel();
            Message gmailmsg = GetMessageDetail(id);
            emailmsg.MessageId = id;
            IList<MessagePartHeader> headers = gmailmsg.Payload.Headers;

            MessagePartHeader fromEmailMPH = (from MessagePartHeader header in headers
where header.Name == "From" select header).FirstOrDefault();
            string fromEmail = (fromEmailMPH == null ? "" : fromEmailMPH.Value);
            emailmsg.From = fromEmail;
            MessagePartHeader subjectMPH = (from MessagePartHeader header in headers
where header.Name == "Subject" select header).FirstOrDefault();
            string subject = (subjectMPH == null ? "" : subjectMPH.Value);
            emailmsg.Subject = subject;
            //for replied to emails
            MessagePartHeader repliedMPH = (from MessagePartHeader header in headers
where header.Name == "In-Reply-To" select header).FirstOrDefault();
            string replied = (repliedMPH == null ? "" : repliedMPH.Value);
            emailmsg.Replied = replied;

            //
            StringBuilder sb = new StringBuilder();
            IList<MessagePart> messageParts = gmailmsg.Payload.Parts;
            string decodedBody = "";
           if (messageParts != null)
            {
                foreach (MessagePart mp in messageParts)
                {

                   string body = mp.Body.Data;
                    if (body != null)
                    {
                        decodedBody = Base64Decode(body);
                    }

                    sb.Append(decodedBody);
                    Console.WriteLine("decodeBody - {0}", decodedBody);
                }

            }
           else
           {
               sb.Append(gmailmsg.Snippet);
           }
           emailmsg.Body = sb.ToString();

            return emailmsg;

        }
```

Figure 18. Code Snippet for Building the Email Message

6. EXPERIMENT EVALUATION AND ANALYSIS

For the experiment, my personal Gmail email corpus was used. Total number of email messages in the corpus were 2776. Out of 2776 emails, nearly 1650 emails were used for the classification experiment.

Table 1. Example of Email Types per Category

Category	**Email Titles / Types**
Shopping Domain	PotteryBarnKids, Claire's, Ann Taylor, Loft, Justice, Gap, Gymboree
Social Media	Invites from LinkedIn, Messages for Facebook
Appointments	West Orthodontics, Friendly Smiles, Evite
Parenting	Mails from Bennett, Elevate Rock School
Personal	Mails from friends
Travel	Mails related to itinerary, Flight Details

Table 2. Results of Email Classification

Categories	**No of Emails Classified Correctly (CCE)**	**No of Emails Misclassified (ME)**	**Total No of Emails Classified (TNE)**	**Accuracy (CCE / TNE) *100**	**Error Rate (ME/TNE) *100**
Appointmens	64	17	81	79.01%	20.98%
Parenting	116	17	133	87.218%	12.789%
Personal	265	43	308	86.03%	13.96%
Shopping Domain	736	19	755	97.48%	2.516%
Social Media	292	10	302	96.68%	3.311%
Travel	53	1	54	98.14%	1.851%

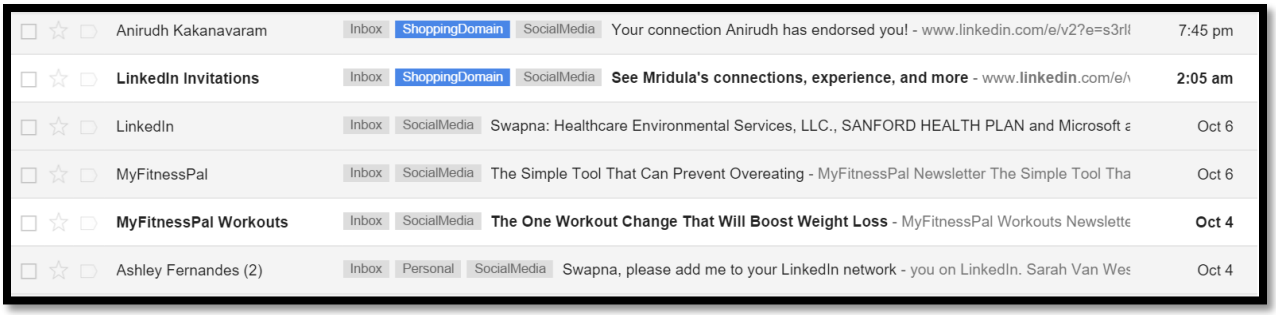

Figure 19. An Example of Misclassified Emails

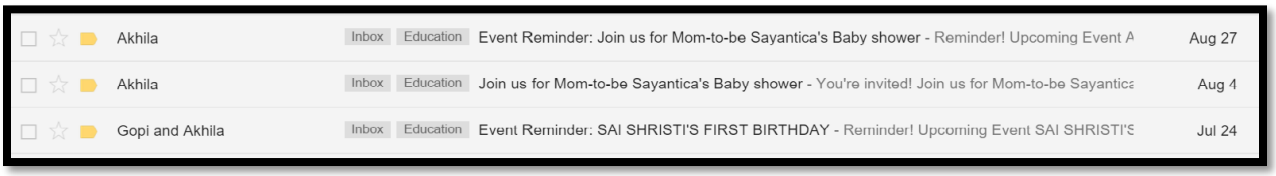

Figure 20. Another Example of Misclassified Emails

In Figure (19), it can be seen that the emails like LinkedIn Invites which should belong to the Social Media category were also classified as belonging to the ShoppingDomain category. Figure (20) shows that the Evite invitations that should be part of Appointments category were classified into the Education category.

Table 3. Average Experiment Results

No of Emails Classified Correctly (CCE)	No of Emails Misclassified (ME)	Total No of Emails for Classification (TNE)	Accuracy	Error Rate
1526	107	1633	93.26%	6.552%

The evaluation is done using below formula:

Accuracy = (CCE / TNE) * 100
= (1526/1633) *100
= 93.26%

Error Rate = (ME/TE) * 100
= (107/1633) * 100
= 6.552%

By analyzing the results of the experiment and looking in detail into the misclassified emails, it was understood that the reason of misclassification was mainly, ambiguity in the words that were common for some categories. For example, the word ‘online’ was part of the Shopping Domain category in the keyword dictionary. When an email that belonged to the appointment category was classified, it contained the word ‘online booking’, and it found a match with the ‘online’ keyword of the Shopping domain category and was misclassified as belonging to Shopping Domain.

7. CONCLUSION AND FUTURE WORK

Taking into account the importance of Electronic messages, in this paper I have proposed an approach of email classification which utilizes the user preferences and adds to the existing knowledge base using self-learning. The main advantage of this approach is reduced manual effort for the user. The algorithm was implemented using ASP.NET C# in Visual Studio 2013 for Gmail messages. The implementation results were studied, and the rules were applied in instances possible. The results show that the approach used is reasonable and takes away the load of manual intervention of the user.

Some of the future research work can be done on, (1) implementing ranking rules for the emails that are classified under more than one category. The ranking rule will improve the accuracy of the classification and will help in assigning the emails to only one specific category. (2) Generating more rules for classifying personal emails. (3) Applying classification rules to emails with content in different languages.

REFERENCES

[1] Jake D. Brutlag and Christopher Meek. “Challenges of the Email Domain for Text Classification,” in *Proc. of the 17th Int. Conf. on Machine Learning,* San Francisco, CA, 2000, pp. 103-110.

[2] William W. Cohen. “Learning Rules that Classify Email,” in *Proc. of the AAAI Spring Symposium on Machine Learning and Information Access,* 1996.

[3] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. “A Bayesian Approach to Filtering Junk E-mail,” in *AAAI-98 Workshop on Learning for Text Categorization,* Madison, WI, 1998, pp. 55-62.

[4] Sebastiani F. “Machine Learning in Automated Text Categorization,” in *ACM Computing Surveys,* Vol. 34, No. 1, 2002, pp. 1-47.

[5] Pratiksha Y. Pawar and S. H. Gawande. “A Comparative Study on Different Types of Approaches to Text Categorization,” in *Int. Journal of Machine Learning and Computing,* vol. 2, no. 4, 2012, pp. 423-426.

[6] Zhongjian Wang, Zongjie Wang, Yanfeng Gao and Yanfen Lin. “Algorithm of E-mail Classification Based on Automatic Adapting for User,” in *Int. Journal of u- and e- Service, Science and Technology,* vol.8, no.2, 2015, pp. 235-242.

[7] Ron Bekkerman, Andrew McCallum, Gary Huang. “Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora,” University of Massachusetts Amherst, Tech. Rep. IR-418, 2004.

[8] Urszula Boryczka, Barbara Probierz, Jan Kozak. “An Ant Colony Optimization Algorithm for an Automatic Categorization of Emails,” in *Computational Collective*

Intelligence. Technologies and Applications, Switzerland: Springer International Publishing, 2014, pp. 583-592.

[9] Neeti Saxena, Bharati Verma, Nitin Shukla. "Online Email Classification Using Ant Clustering Algorithm," in *Int. Journal of Emerging Technology and Advanced Engineering,* vol. 2, 2012.

[10] Manu Aery, Sharma Chakravarthy. "eMailSift: Email Classification based on Structure and Content," in *Proc. of the 5th IEEE Int. Conf. on Data Mining (ICDM05),* Clearwater Beach, FL, 2005, pp.18-25.

[11] Denil Vira, Pradeep Raja & Shidharth Gada. "An Approach to Email Classification Using Bayesian Theorem," in *Global Journal of Computer Science and Technology Software and Data Engineering,* vol. 12, 2012.

[12] Richard B. Segal and Jeffrey O. Kephart. "SwiftFile: An Intelligent Assistant for Organizing E-Mail," in *AAAI 2000 Spring Symposium on Adaptive User Interfaces,* Stanford, CA, 2005. Doi: 10.1109/ICDM.2005.58.

[13] Richard B. Segal and Jeffrey O. Kephart. "MailCat: An Intelligent Assistant for Organizing E-Mail," in *Proc. of the 3rd Annu. Conf. on Autonomous Agents,* 1999, pp. 276-282.

[14] Jason D. M. Rennie. "ifile: An Application of Machine Learning to E-Mail Filtering," in *Proc. of the KDD- 2000 Workshop on Text Mining,* Boston, MA, 2000.

[15] K. Saruladha and L.Sasireka. "Survey of text classification algorithms for Spam Filtering," in *Int. Journal of Innovative Trends in Engg.*, 2012, pp. 233-237.

[16] Seongwook Youn, Dennis McLeod. "Spam Email Classification using an Adaptive Ontology," in *IEEE Int.* Conf. on Information Technology (ITNG'07), pp. 249-254, 2007.

[17] B. Cui, A. Mondal, J. Shen, G. Cong, and K. Tan. “On Effective Email Classification via Neural Networks,” in *Proc. of the Database and Expert Systems Applications,* vol. 3588, 2005, pp. 85-94.

[18] Wikipedia. (2002). *Linear Classifier* [Online]. Available: https://en.wikipedia.org/wiki/Linear_classifier [Accessed October 2015].

[19] Archit Mehta, Raunakraj Patel, Jatin Savaliya. “Email Classification using Data Mining,” Bachelors Project, Dept. Comp. Eng., Sardar Patel University, Gujrat, India.

[20] Akron-Summit County Public Library. (2006). *E-Mail Basics Part 1* [Online]. Available: http://www.akronlibrary.org/training/pdf/Email_Part_1.pdf [Accessed October 2015].

[21] D Bnonn Tennant. (2012). *5 reasons Email Marketing Crushes Social Media Marketing for B2B* [Online]. Available: https://blog.kissmetrics.com/email-crushes-social-media/ [Accessed October 2015].

[22] *What’s in Visual Studio Community 2013* [Online]. Available: https://www.visualstudio.com/en-us/news/vs2013-community-vs.aspx [Accessed October 2015].

[23] Google Apps Team (2014). *Gmail API Overview* [Online]. Available: https://developers.google.com/gmail/api/?hl=en [Accessed October 2015].

[24] CCM.net-Kioskea (2015). *Structure of an email (headers and bodies)* [Online]. Available: http://ccm.net/contents/117-structure-of-an-email-headers-and-bodies. [Accessed October 2015].